Poster: Clean-label Backdoor Attack on Graph Neural Networks

Jing Xu
Delft University of Technology
Delft, Netherlands
j.xu-8@tudelft.nl

Stjepan Picek
Radboud University & Delft University of Technology
Nijmegen, Netherlands
stjepan.picek@ru.nl

ABSTRACT

Graph Neural Networks (GNNs) have achieved impressive results in various graph learning tasks. They have found their way into many applications, such as fraud detection, molecular property prediction, or knowledge graph reasoning. However, GNNs have been recently demonstrated to be vulnerable to backdoor attacks. In this work, we explore a new kind of backdoor attack, i.e., a clean-label backdoor attack, on GNNs. Unlike prior backdoor attacks on GNNs in which the adversary can introduce arbitrary, often clearly mislabeled, inputs to the training set, in a clean-label backdoor attack, the resulting poisoned inputs appear to be consistent with their label and thus are less likely to be filtered as outliers. The initial experimental results illustrate that the adversary can achieve a high attack success rate (up to 98.47%) with a clean-label backdoor attack on GNNs for the graph classification task. We hope our work will raise awareness of this attack and inspire novel defenses against it.

CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies** → **Machine learning**;

KEYWORDS

Backdoor Attacks; Graph Neural Networks; Graph Classification

ACM Reference Format:
Jing Xu and Stjepan Picek. 2022. Poster: Clean-label Backdoor Attack on Graph Neural Networks. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS '22), November 7–11, 2022, Los Angeles, CA, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3548606.3563531

1 INTRODUCTION

Graph Neural Networks (GNNs) have made tremendous progress in various domains, e.g., drug design [8], fake news detection in social media [15], recommendation system [9], and even financial sector [7]. However, similar to Convolutional Neural Networks (CNNs), it has been demonstrated that GNNs are vulnerable to backdoor attacks. For instance, in a Bitcoin transaction ego network [4], where nodes are the transactions, and the edge between two nodes indicates the flow of Bitcoin from one transaction to another, the adversary can attack the GNNs to classify an illegal transaction as a benign one.

Like prior backdoor attacks on CNNs, recent studies about backdoor attacks on GNNs assume the adversary can often introduce mislabeled inputs to the training dataset [11, 12, 16]. Specifically, the attacker modifies the training dataset by injecting a backdoor trigger into some training samples and relabeling these samples to the chosen target label. Then the backdoored neural network classifier will output the attacker-chosen label when a trigger is injected into a testing sample.

However, it has been demonstrated for CNNs that even a fairly simple filtering process will detect the poisoned samples as outliers [6]. More importantly, any subsequent human inspection will deem these inputs suspicious and thus potentially reveal the attack [6]. To make the resulting poisoned inputs appear consistent with their labels so that it is more difficult to detect the poisoned inputs, a clean-label backdoor attack was proposed [6]. In the clean-label backdoor attack, the adversary only poisons inputs of the target class without changing the true labels. The backdoored model aims to predict the testing sample with a trigger into the target label.

Although graphs are difficult to visualize directly for humans, unlike images and texts [14], there are still some straightforward methods to detect poisoned graph samples. For instance, we can apply the GNN prediction explanation tool, e.g., GNNExplainer [14], to visualize semantically relevant graph structures that are interpretable for humans. Specifically, we first get explanation subgraphs for each class, and then for each graph sample, check whether it contains the corresponding class explanation subgraphs. If a graph sample in a specific class does not contain the corresponding class explanation subgraphs, we can consider it an outlier. Therefore, it is also crucial to study the clean-label backdoor attacks on GNNs. So far, clean-label backdoor attacks on deep neural networks have been proposed in various domains, e.g., image classification [6] and video recognition [17]. Still, to the best of our knowledge, there is no work on the clean-label backdoor attacks on GNNs. We aim to bridge this gap. This work explores the clean-label backdoor attack on GNNs. More specifically, we aim to investigate whether using clearly mislabeled graphs is necessary for implementing a backdoor attack on GNNs. Can we carry out such backdoor attacks by insisting that each poisoned graph and its label must be consistent? Our preliminary results show that a high attack success rate can be achieved even with a clean-label backdoor attack on GNNs.

2 METHODOLOGY

2.1 Problem Formulation

GNNs take a graph $G = (V, E, X)$ as an input, where V, E, X denote nodes, edges, and node attributes, and learn a representation vector

CCS '22, November 7–11, 2022, Los Angeles, CA, USA

ACM ISBN 978-1-4503-9450-5/22/11.
https://doi.org/10.1145/3548606.3563531

(embedding) for each node, z_v ($v \in G$), or the entire graph, z_G. Modern GNNs update the representation of a node by aggregating representations of its neighbors. After k iterations of aggregation, a node's representation captures both structure and feature information within its k-hop network neighborhood. For the node classification task, the node representation z_v is used for prediction. For the graph classification task, the READOUT function pools the node representations for a graph-level representation z_G. The graph classification task aims to predict the class label(s) for an entire graph using the graph-level representation z_G.

Given a pre-trained GNN model Φ_o and its training dataset $D_{train} = \{(G_1, y_1), (G_2, y_2), \ldots, (G_n, y_n)\}$ where G_i and y_i respectively are the i-th training graph and its true label, the clean-label backdoor attack aims to forge a backdoored GNN Φ_b that will misclassify the testing sample with a specific trigger into predetermined labels (i.e., target label y_t) without affecting the performance on clean data. We assume the attacker can access the training dataset D_{train}. Unlike prior works [11, 12, 16], which perform backdoor attacks on GNNs by injecting a trigger into a sampled training dataset and changing their labels to the target label, the attacker of clean-label backdoor attack samples a subset of training dataset with target label and injects trigger into them without changing their labels. Thus, the poisoned samples have plausible labels.

2.2 General Framework

The general framework of a clean-label backdoor attack on GNNs is shown in Figure 1. In the training phase, as presented in Figure 1a, the attacker samples data from the original training dataset in the target class and injects a specific trigger to generate poisoned samples. The resulting poisoned samples are then utilized to backdoor the pre-trained GNN model Φ_o to get the backdoored GNN model Φ_b. Here, we focus on the subgraph-based backdoor attacks on GNNs for the graph classification task since most graph classification tasks are implemented in GNNs by learning the network structure. The backdoored GNN model is assumed to predict any testing sample (which can be from an untarget class) with a specific trigger into the target class, as shown in Figure 1b.

Specifically, the implementation details of our attack are described in Algorithm 1. The key point is backdoored dataset generation. Here, we adopt the Erdős-Rényi (ER) model [2] to generate trigger g_t (line 3 in Algorithm 1) as it is fast and more effective than other random graph generation methods [16]. In particular, this model outputs a random graph of s nodes, and the probability of an edge between each pair of nodes in this graph is ρ. We sample subsets of the original training dataset (with target label) with proportion r, as shown as D_{tmp}, and the remaining is saved as clean training dataset D_{clean}. For each sampled graph, we inject a trigger (by the ER model) into it by sampling s nodes from the graph uniformly at random and replacing their connection with that in the trigger graph. Under the setting of a clean-label backdoor attack, the attacker does not re-label the sampled data. The backdoored dataset comprises the dataset with trigger $D_{trigger}$ and the remaining clean training dataset D_{clean}.

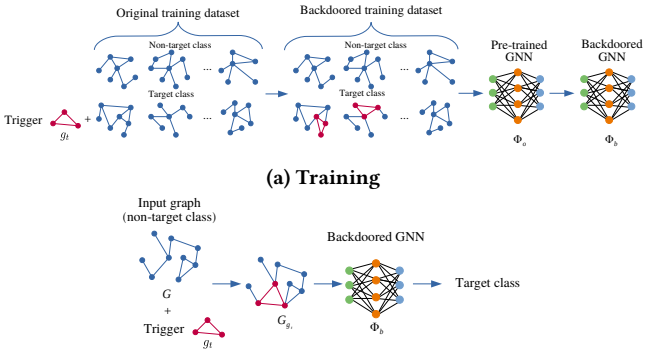

Figure 1: Clean-label backdoor attack framework.

Algorithm 1: Clean-label backdoor attack for the graph classification task

Input: Pre-trained GNN model Φ_o, Training set $D_{train} = \{x_i, y_i\}_{i=1}^{S}$, Target label $y_t \in [0, C)$
Output: Backdoored GNN model Φ_b

1 **Function** CLEAN_LABEL_BACKDOOR_ATTACK():
2 $\quad D_{trigger} \leftarrow \emptyset$
3 $\quad g_t \leftarrow ER(s, \rho)$
4 $\quad D_{tmp} \leftarrow sample(D_{train}, r, y = y_t)$
5 $\quad D_{clean} = \{data \in D_{train} : data \notin D_{tmp}\}$
6 $\quad$ **foreach** $d \in D_{tmp}$ **do**
7 $\quad\quad d[x] = INJECT(d[x], g_t)$
8 $\quad\quad D_{trigger} = D_{trigger} \cup \{d[x], d[y]\}$
9 $\quad$ **end**
10 **End Function**
11 $D_{backdoor} = D_{clean} \bigcup D_{trigger}$
12 $\Phi_b = Train(\Phi_o, D_{backdoor})$
13 **return** Φ_b

3 EXPERIMENTAL RESULTS

Datasets. We perform experiments with two publicly available datasets: (1) MUTAG [1] - structure graphs of the mutagenic and non-mutagenic molecules and (2) NCI1 [5] - chemical compounds screened for activity against non-small cell lung cancer and ovarian cancer cell lines. For each dataset, we randomly sample 80% of the data instances as the training dataset and the rest as the testing dataset.

Target Models. We choose GCN [3] and GIN [13] as our target models to be attacked, considering their excellent performance and widespread adoption [8, 10].

Evaluation. We use the *attack success rate (ASR)* to evaluate the attack effectiveness. Specifically, we embed each testing data with the specific trigger graph and calculate the ASR of the backdoored GNN model on the poisoned testing dataset. Here, we only embed the testing dataset of the non-target label with a trigger to avoid the influence of the original label. The ASR measures the proportion of trigger-embedded inputs misclassified by the backdoored GNN into the target class y_t chosen by the adversary. The trigger-embedded

inputs are

$$D_{trigger} = \left\{(G_{1,g_t}, y_1), (G_{2,g_t}, y_2), \ldots, (G_{n,g_t}, y_n)\right\}.$$

Here, g_t is the graph trigger, $\left\{G_{1,g_t}, G_{2,g_t} \ldots, G_{n,g_t}\right\}$ is the testing dataset embedded with g_t, and $y_1, y_2, \ldots, y_n$ is the label set.

Formally, ASR is defined as:

$$Attack\ Success\ Rate = \frac{\sum_{i=1}^{n} \mathbb{I}(\Phi_b(G_{i,g_t}) = y_t)}{n},$$

where $\mathbb{I}$ is an indicator function and Φ_b refers to the backdoored GNN model.

Furthermore, we also use *clean accuracy drop (CAD)* to evaluate the attack evasiveness. CAD indicates the classification accuracy difference between the original GNN model Φ_o and the backdoored GNN model Φ_b over the clean testing dataset.

Results. We set the number of nodes in the trigger graph to be 20% of the average number of nodes in the dataset. Next, we set the poisoning rate r and edge existing probability ρ to be 10% and 80%, respectively. We set those parameters following the setting in prior backdoor attacks on GNNs [12, 16]. Table 1 presents the attack results. We can observe that overall, a clean-label backdoor attack can achieve high attack effectiveness for both datasets and models, i.e., with an attack success rate over 84%, especially for the NCI1 dataset with up to 98.47% ASR. It can also be observed that in most cases, a clean-label backdoor attack has a low CAD, i.e., around 1%, which indicates that a clean-label backdoor attack has a negligible impact on the original task of the model.

The results in [6] indicate that only poisoning inputs of the target class (i.e., without changing the true labels) renders the attack ineffective. The authors argued the main reason for the attack's ineffectiveness is that the poisoned samples can be correctly classified by learning a standard classifier, so the backdoor attack is unlikely to be successful since relying on the backdoor trigger is not necessary to classify these inputs correctly. On the contrary, here, we find that without any other improvement, the clean-label backdoor attack is already successful on GNN models. This may be explained since the GNN model predicts the input graph by learning information of specific structure(s), i.e., explanation subgraph(s) [14], in the graph. If a graph is injected with a trigger graph, the GNN model will also try to learn the trigger pattern and add it to the explanation subgraphs. Once the backdoored GNN model is trained, it will output a target label if one of the explanation subgraphs, e.g., trigger graph, appears in the input graph.

Table 1: Attack performance (SD: standard deviation).

Dataset	ASR(%) | CAD(%) Mean (SD)	
	GCN	GIN
MUTAG	87.83(1.03) | 0.16(0.03)	84.86(1.68) | 2.24(0.15)
NCI1	98.47(1.30) | 0.88(0.01)	97.62(2.03) | 1.01(0.07)

4 CONCLUSIONS AND FUTURE WORK

The original backdoor attacks on GNNs [11, 12, 16] crucially relies on the addition of arbitrary, most mislabeled inputs into the training set. This raises the risk that the poisoned, mislabeled inputs will likely be detected by filtering or some GNN explanation tools. We argue that, for a backdoor attack to be insidious, the attacker must not rely on inputs that appear mislabeled. Thus, in this work, we explore a new backdoor attack on GNNs that only poisons inputs of the target class without changing the true labels. Our method leverages the GNN model's redundant learning capability to learn the trigger pattern. Initial experimental evaluations showed that a clean-label backdoor attack could achieve a high attack success rate and low clean accuracy drop. We hope our study highlights the concern of clean-label backdoor attacks on GNNs, which are more insidious.

In future work, we aim to explore our method's effectiveness against simple filtering techniques mentioned in Section 1. Additionally, it will be interesting to compare the attack results of former backdoor attacks and clean-label backdoor attacks on GNNs against possible defenses.

5 ACKNOWLEDGMENTS

This work is supported by the China Scholarship Council (CSC).

REFERENCES

[1] Asim Kumar Debnath, Rosa L Lopez de Compadre, Gargi Debnath, Alan J Shusterman, and Corwin Hansch. 1991. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of medicinal chemistry* 34, 2 (1991).

[2] E. N. Gilbert. 1959. Random Graphs. *The Annals of Mathematical Statistics* 30, 4 (1959), 1141–1144. https://doi.org/10.1214/aoms/1177706098

[3] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[4] Dániel Kondor, Márton Pósfai, István Csabai, and Gábor Vattay. 2014. Do the rich get richer? An empirical analysis of the Bitcoin transaction network. *PloS one* 9, 2 (2014), e86197.

[5] Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. 2011. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research* 12, 9 (2011).

[6] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. 2018. Clean-label backdoor attacks. (2018).

[7] Daixin Wang, Jianbin Lin, Peng Cui, Quanhui Jia, Zhen Wang, Yanming Fang, Quan Yu, Jun Zhou, Shuang Yang, and Yuan Qi. 2019. A semi-supervised graph attentive network for financial fraud detection. In *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 598–607.

[8] Lingfei Wu, Peng Cui, Jian Pei, Liang Zhao, and Le Song. 2022. Graph neural networks. In *Graph Neural Networks: Foundations, Frontiers, and Applications*. Springer, 27–37.

[9] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 346–353.

[10] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* 32, 1 (2020), 4–24.

[11] Zhaohan Xi, Ren Pang, Shouling Ji, and Ting Wang. 2021. Graph backdoor. In *30th USENIX Security Symposium (USENIX Security 21)*. 1523–1540.

[12] Jing Xu, Minhui Xue, and Stjepan Picek. 2021. Explainability-based backdoor attacks against graph neural networks. In *Proceedings of the 3rd ACM Workshop on Wireless Security and Machine Learning*. 31–36.

[13] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826* (2018).

[14] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems* 32 (2019).

[15] Jiawei Zhang, Bowen Dong, and S Yu Philip. 2020. Fakedetector: Effective fake news detection with deep diffusive neural network. In *2020 IEEE 36th international conference on data engineering (ICDE)*. IEEE, 1826–1829.

[16] Zaixi Zhang, Jinyuan Jia, Binghui Wang, and Neil Zhenqiang Gong. 2021. Backdoor attacks to graph neural networks. In *Proceedings of the 26th ACM Symposium on Access Control Models and Technologies*. 15–26.

[17] Shihao Zhao, Xingjun Ma, Xiang Zheng, James Bailey, Jingjing Chen, and Yu-Gang Jiang. 2020. Clean-label backdoor attacks on video recognition models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14443–14452.